Une méthode statistique robuste à deux niveaux pour le recalage temporel à partir de primitives de type différent

Gilles Simon, Marie-Odile Berger

Une méthode statistique robuste à deux niveaux pour le recalage temporel à partir de primitives de type différent

A two-stage robust statistical method for temporal registration from features of various type

G. Simon M.-O. Berger
CRIN-CNRS & INRIA Lorraine, Bâtiment LORIA, BP 239,
54506 Vandœuvre-lès-Nancy cedex, France
e-mail:{gsimon,berger}@loria.fr

Résumé

Nous présentons un système de recalage temporel 3D/2D, capable de suivre dans une séquence d'images un objet 3D dont le modèle est connu. Il comporte le suivi des primitives, le calcul du point de vue et la mise à jour des primitives visibles. Le coeur de notre système est la méthode de calcul du point de vue, qui prend en compte de façon très robuste des primitives de type différent (points, droites et courbes 3D non nécessairement définies paramétriquement) et est capable de donner une estimation correcte du point de vue même lorsque des erreurs de suivi se produisent. Nous montrons la fiabilité de notre système dans le cadre d'un projet de réalité augmentée.

Mots Clef

Calcul du point de vue, estimation robuste, recalage temporel, réalité augmentée, suivi.

Abstract

A model registration system capable of tracking an object, the model of which is known, in an image sequence is presented. It integrates tracking, pose determination and updating of the visible features. The heart of our system is the pose computation method, which handles various features (points, lines and free-form curves) in a very robust way and is able to give a correct estimate of the pose even when tracking errors occur. The reliability of the system is shown on an augmented reality project.

Keywords

Pose computation, robust estimation, temporal registration, augmented reality, tracking.

1 Introduction

Notre objectif est de construire un système de recalage 3D/2D, capable de suivre un objet 3D de modèle connu dans une séquence d'images (le modèle de projection utilisé étant la projection perspective). Un tel système est d'un grand intérêt pour les applications de réalité augmentée, en particulier lorsque des objets virtuels doivent être incrustés dans une séquence d'images, ou lorsqu'un objet de la scène doit être remplacé par un autre objet [2, 18, 21].

Le système de recalage est initialisé par un ensemble de correspondances 3D/2D dans la première image, utilisées pour estimer le point de vue initial, et les paramètres intrinsèques de la caméra sont supposés connus. Une fois initialisé, le système doit être capable de suivre automatiquement les primitives dans les images suivantes, et de calculer le point de vue (position et orientation de la caméra par rapport à l'objet 3D) à partir de leur correspondance avec le modèle 3D. Puisque les primitives pertinentes qui apparaissent dans une image sont souvent des contours courbes (notamment dans les scènes d'extérieur), notre méthode de calcul du point de vue doit pouvoir prendre en compte aussi bien des points et des droites que des courbes 3D quelconques.

L'algorithme de suivi permet le plus souvent de déterminer les primitives homologues. Il se peut néanmoins que des erreurs de suivi se produisent et que certaines primitives du modèle soient appariées avec des primitives images incorrectes. Une seule erreur peut avoir de lourdes conséquences sur le calcul du point de vue. Pour des primitives de type point, une approche robuste permet de déterminer si le point est erroné ou non [10]. Lorsqu'on considère des primitives courbes, le problème est plus complexe car certaines parties de

la primitive 2D peuvent parfaitement correspondre au modèle 3D, alors que d'autres parties peuvent être fausses (voir par exemple la primitive 4 de la figure 6.b) : il est donc nécessaire d'utiliser un algorithme capable d'extraire les parties des primitives qui correspondent au modèle 3D.

En plus du problème du calcul du point de vue, d'autres problèmes doivent être résolus pour maintenir le recalage au cours du temps : comme la caméra effectue un mouvement par rapport à l'objet, de nouvelles primitives peuvent apparaître tandis que des primitives qui jusqu'ici étaient visibles peuvent disparaître. L'ensemble des primitives du modèle suivies dans la séquence doit donc être dynamiquement mis à jour. Cela signifie que nous devons déterminer aussi bien les primitives 3D nouvellement visibles, que les correspondants 2D qui vont être suivis.

Etat de l'art

La majorité des algorithmes de calcul du point de vue, basés sur la connaissance de correspondances 3D/2D, adoptent la démarche suivante : des hypothèses d'appariements sont d'abord générées, puis le point de vue est calculé en utilisant ces hypothèses. Dans le contexte du suivi, une estimation grossière du point de vue étant disponible (le point de vue calculé dans l'image précédente), le correspondant peut être recherché dans un voisinage limité de la primitive projetée.

Puisqu'une seule erreur grossière peut fausser le calcul du point de vue, on porte souvent une attention toute particulière à l'étape d'appariement. Par exemple, Lowe [16] utilise un critère probabiliste pour guider la recherche des correspondances, avant d'utiliser une estimation aux moindres carrés pondérée incluant des contraintes *a priori* pour la stabilisation. D'autres méthodes [9, 13] utilisent un modèle de vitesse et un filtre de Kalman pour prédire la position de la primitive image. Malheureusement, l'utilisation d'un modèle de vitesse impose des contraintes de régularité sur le mouvement de la caméra, ce qui peut s'avérer inapproprié pour des applications de réalité augmentée où la scène est souvent filmée par un observateur en mouvement.

Nous proposons donc une approche moins contraignante : au lieu d'essayer d'affiner l'étape d'appariement, nous préférons utiliser une méthode statistique robuste pour calculer le point de vue à partir des appariements issus de l'étape du suivi. Le résultat que nous obtenons est une estimation robuste du point de vue, ainsi que les parties des primitives suivies qui ne correspondent pas au modèle. De plus, ceci nous permet de mettre à jour très facilement l'ensemble des primitives suivies.

Notre méthode se rapproche de Ravela *et al.* [18], qui utilisent des techniques robustes pour calculer le point de vue après l'étape du suivi. Cependant, ces travaux ne considèrent que des points alors que nous sommes capables de prendre en compte des primitives quelconques. Une autre différence fondamentale est la façon de mettre à jour l'ensemble des primitives suivies : dans [18], un graphe d'aspect est utilisé pour étiqueter les points qui sont visibles pour un angle de vue donné, mais ces points sont extraits à la main. De plus, la stratégie de suivi utilisée, basée sur la corrélation, ne peut être étendue à des courbes quelconques.

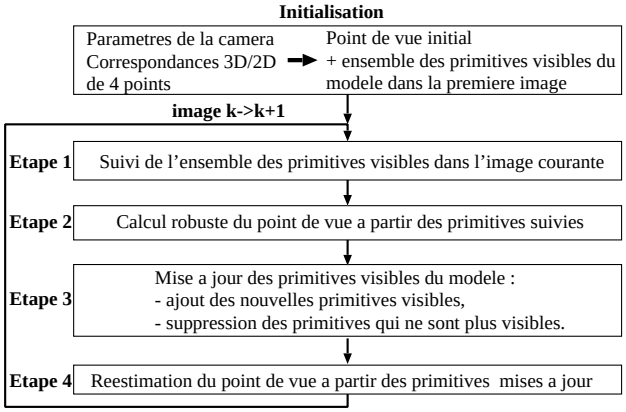


FIG. 1: *La boucle de recalage temporel.*

2 Vue d'ensemble du système

La phase d'initialisation de notre système comporte les étapes suivantes:

- les paramètres internes de la caméra sont déterminés par une calibration classique [6] ;
- le modèle étant complexe, on extrait un ensemble de primitives 3D susceptibles d'apparaître de façon pertinente dans la séquence ;
- l'opérateur désigne dans la première image quatre points dont les correspondants 3D sont connus. Ces quatre points sont utilisés pour initialiser le point de vue (par la méthode de Dementhon et Davis [4]) et ainsi déterminer les primitives 3D visibles dans la première image. Les primitives 2D correspondantes sont alors déterminées de façon automatique en utilisant la méthode de mise à jour décrite plus bas.

Une fois initialisé, le système parcourt une boucle comprenant quatre étapes, résumées en figure 1 :

Etape 1 : Suivi des primitives. L'ensemble des primitives est suivi dans l'image courante en utilisant un algorithme de suivi de courbes que nous avons

précédemment développé [1] : une estimation grossière du champ des vitesses est d'abord calculée sur le contour suivi, en modélisant ce champ par un déplacement 2D rigide. On peut alors calculer une prédiction de la localisation du contour dans l'image suivante. En utilisant les contours actifs à partir de cette prédiction, le contour correspondant peut être détecté. Nous avons montré que notre méthode est fiable, rapide et capable de prendre en compte des mouvements suffisamment grands. Cependant, le suivi peut quelquefois échouer si le *snake* est attiré par des maxima de gradient locaux : ceci se produit souvent lorsqu'une courbe n'est que partiellement correctement localisée (voir les primitives 1 et 4 de la figure 6.d). Si l'étape de prédiction échoue (en raison de l'instabilité du calcul du flot), la position obtenue est généralement complètement fausse (voir la primitive 5 de la figure 6.d). Ainsi, parmi l'ensemble des contours suivis, un petit nombre d'entre eux peuvent être mal détectés ou complètement erronés.

Etape 2 : Calcul du point de vue. Cette étape est le coeur de notre système. Elle permet de calculer le point de vue à partir de primitives de type différent (points, droites, courbes 3D) malgré les éventuelles erreurs de suivi. Nous utilisons pour cela un algorithme de type *Iterative Closest Point* (ICP). Notre méthode se distingue des travaux précédents par l'utilisation d'estimateurs robustes dans un processus à deux niveaux : un **niveau local**, qui calcule un résidu robuste pour chaque primitive, et un **niveau global** qui minimise une fonction robuste de ces résidus. L'intérêt de cet algorithme est double : d'une part, il permet de calculer le point de vue de façon robuste en n'utilisant que les parties des primitives 2D correspondant au modèle 3D, et d'autre part il permet de détecter les primitives erronées.

Etape 3 : Mise à jour des primitives visibles. Lorsque la caméra balaye un champ large (mouvement panoramique par exemple), il devient essentiel de mettre à jour l'ensemble $\mathcal{S}$ des primitives suivies. Dans le cas contraire, le nombre de primitives visibles décroît (jusqu'à éventuellement devenir nul) et le point de vue ne peut plus être calculé de façon fiable. Pour chaque primitive théoriquement visible dans l'image et non encore utilisée, on considère tous les contours qui sont suffisamment proches de la projection de la primitive 3D. Pour chacun de ces contours c, on calcule le point de vue correspondant à $\mathcal{S} \cup c$ en utilisant l'algorithme de l'étape 2, et on regarde si le contour c est éliminé ou non : s'il ne l'est pas, cela signifie que c est susceptible d'être le correspondant de la primitive modèle et est ajouté à $\mathcal{S}$.

Etape 4 : Réestimation du point de vue. Pour finir, le point de vue est réestimé en tenant compte de la mise à jour de $\mathcal{S}$.

Les sections suivantes détaillent chacune de ces étapes. Des résultats sur une application de réalité augmentée sont montrés en section 6.

3 Suivi des primitives

Le suivi de primitives courbes est souvent une tâche difficile. Il ne peut être basé sur la notion de contour car la courbe à suivre est souvent la réunion de plusieurs chaines de contours et le suivi passe alors par une phase délicate de découpage/fusion des contours entre deux images consécutives. L'utilisation d'une mise en correspondance globale par le biais des contours actifs [12] est alors très intéressante car elle permet de résoudre en même temps le problème de la segmentation et celui du suivi: à partir d'un contour initial, la courbe va converger vers le contour le plus proche dans l'image suivante. Malheureusement, cette méthode est inefficace si les courbes en correspondance sont éloignées.

Notre méthode de suivi inclut donc une étape de prédiction destinée à fournir une initialisation correcte au processus des contours actifs. Ceci revient à estimer le champs des vitesses 2D entre les images. En raison de la flexibilité des contours actifs, une estimation grossière est suffisante et nous avons modélisé le champ des vitesses $\mathcal{M}$ sur le contour à suivre par un déplacement rigide 2D noté $\mathcal{D}$.

Le calcul du flot optique complet est très sensible au bruit.Le flot normal est par contre plus facile à calculer mais il n'est qu'une approximation du déplacement entre les deux images, surtout si ce déplacement est important. Nous avons donc opté pour une détermination itérative du champ à partir du flot optique normal; l'idée sous jacente est de calculer une première estimation du champ compatible avec le flot normal sur le contour puis de recaler les deux images en utilisant le déplacement calculé. Les contours en correspondance sont alors plus proches et le flot optique va donc être plus précis. En itérant ce processus, on parvient petit à petit à amener le contour de la première image sur celui qui lui ressemble le plus dans l'image suivante. Plus formellement:

Soit $(M_i)_{0 \leq i < N}$ la discrétisation de la courbe à suivre $\mathcal{C}$ et soit $f_0^{\perp}(M_i)$ le flot optique normal au point (M_i) (la normale à la courbe au point (M_i) est notée n_i). On note I_1 et I_2 deux image consécutives de la séquence.

$\mathcal{D}_0$ est le déplacement rigide 2D minimisant

$$\sum_{0 \leq i < N} |(\overrightarrow{M_i \mathcal{D}_0(M_i)}.n_i)n_i - f_0^{\perp}(M_i)|^2 \qquad (1)$$

Cette estimation est ensuite affinée itérativement en considérant le flot optique normal $f_1^{\perp}$ sur $\mathcal{D}_0(\mathcal{C})$ entre les images $I_1(\mathcal{D}_0^{-1}(x,y))$ et I_2. Des déplacement successifs infinitésimaux $\mathcal{D}_0, ..., \mathcal{D}_j, ...$ sont ainsi calculés et on a $\mathcal{D} = \mathcal{D}_n \ldots \mathcal{D}_1 \mathcal{D}_0$.
L'intérêt de choisir un modèle explicite et global du mouvement sur le contour (ici un modèle rigide 2D) permet d'éviter que des instabilités inévitables dans le calcul du flot normal viennent perturber l'estimation du champ des vitesses. L'approche itérative nous permet de prendre en compte des déplacements de l'ordre d'une vingtaine de pixels entre les deux images.
Il peut cependant arriver que le suivi soit erroné: soit parce que le flot est très bruité sur une grosse partie de la courbe considérée, soit parce que le contour actif a été attiré par un gradient local très fort. Néanmoins, ceci ne concerne en pratique qu'un petit nombre des primitives suivies.

4 Calcul du point de vue

La plupart des algorithmes de calcul du point de vue utilisent des primitives simples : des points [10, 4], des droites [5, 20, 15] ou des cercles [8]. Mais peu de travaux sont consacrés au recalage à partir de courbes 3D quelconques. Kriegman [14] propose une méthode algébrique pour calculer le point de vue à partir de l'observation des contours occultants d'un objet courbe (ceci peut être facilement appliqué au recalage 3D/2D). Malheureusement, cette méthode ne fonctionne qu'avec des surfaces ou des courbes que l'on peut définir de façon paramétrique (fractions de polynômes). De plus, l'utilisation de la théorie de l'élimination s'avère rapidement très coûteuse. Le problème du recalage 3D/2D est aussi considéré dans [7], pour des applications médicales. Partant d'une estimation grossière du point de vue, un algorithme ICP est utilisé pour effectuer le recalage. Un filtre de Kalman étendu est utilisé pour éliminer les erreurs grossières en effectuant des tests de χ^2. Cependant, comme le résultat dépend de l'ordre dans lequel on réalise les mesures, l'estimation risque de n'être qu'un minimum local, en particulier si l'estimée initiale n'est pas très précise. D'autre part, cette méthode est valide sur une courbe, mais aucune solution n'est proposée pour gérer n courbes dont certaines peuvent être complètement fausses.

4.1 Le problème

Le problème consiste à calculer la rotation $\mathbf{R}$ et la translation $\mathbf{t}$ qui permettent de passer du repère du modèle au repère de la caméra. Si les paramètres intrinsèques de la caméra sont connus, il s'agit donc de déterminer six paramètres (trois pour $\mathbf{R}$ et trois pour $\mathbf{t}$), que nous désignerons par le vecteur $\mathbf{p}$. Les correspondances 3D/2D de n primitives quelconques (décrites par des chaînes de points) sont supposées connues. Soit :

- C_i une courbe 3D, décrite par la chaîne de points 3D $\{M_{i,j}\}_{1 \leq j \leq l_i}$ (C_i peut donc être une primitive quelconque, y compris un point ou une droite),
- c_i la projection de C_i dans le plan image, décrite par la chaîne de points 2D $\{m_{i,j}\}_{1 \leq j \leq l_i}$, où $m_{i,j} = Proj(\mathbf{R}M_{i,j} + \mathbf{t})$ ($Proj$ étant la fonction de projection perspective d'un point 3D dans le plan image),
- c_i' le contour détecté dans l'image (le contour suivi) correspondant à C_i, décrit par la chaîne de points 2D $\{m'_{i,j}\}_{1 \leq j \leq l'_i}$.

Une solution simple consisterait à effectuer la minimisation à un seul niveau

$$\min \sum_{i,j} \rho(d_{i,j}) \qquad (2)$$

où $d_{i,j} = Dist(m'_{i,j}, c_i)$ et $Dist$ est une fonction qui approxime la distance euclidienne d'un point à un contour (par une méthode classique d'approximation autour du point le plus proche) et ρ une fonction symétrique et positive, utilisée pour réduire l'influence des parties erronées (voir plus bas).
Malheureusement, cette méthode n'est pas satisfaisante car elle fond toutes les primitives en un ensemble de points et ne distingue pas les erreurs locales (quand une primitive n'est que partiellement correctement localisée) des erreurs grossières (quand la position de la primitive image est complètement fausse). Pourtant, ces deux types d'erreur ne sont pas du tout de même nature, et ne pas les traiter séparément implique une grande perte en robustesse et en précision.
Pour remédier à cela, nous proposons d'utiliser des estimateurs robustes dans un processus à deux niveaux : un **niveau local**, qui calcule un résidu robuste pour chaque primitive, et un **niveau global**, qui minimise une fonction robuste de ces résidus. Le niveau local réduit (ou même supprime) l'influence des sections erronées des contours (voir les primitives 1 et 4 de la figure 6.d), tandis que le niveau global élimine les *primitives aberrantes*, c'est à dire les contours qui sont complètement faux, ou qui contiennent une trop grande proportion de points erronés (voir la primitive 5 de la figure 6.d).

4.2 Le niveau global

Ce niveau fait coïncider la projection des primitives 3D avec les primitives 2D suivies en minimisant les résidus r_i qui sont calculés pour chaque couple de primitives 3D/2D (cf. niveau local). Afin de réduire l'influence des primitives aberrantes, c'est-à-dire les primitives dont le résidu reste relativement grand lorsque le point de vue correct est trouvé, nous devons effectuer une optimisation robuste (si on utilise un simple moindres carrés - LS, ce qui revient à minimiser $\sum_{i=1}^{n} r_i^2$, l'influence des primitives croît linéairement avec la taille de leur résidu).
Les statisticiens ont suggéré de nombreuses méthodes robustes. Parmi elles, les deux plus populaires sont les M-estimateurs et la méthode des *moindres carrés médians* (LMS), qui ont été utilisées pour résoudre de nombreux problèmes de vision [10, 15, 22].
La technique LMS, suggérée par Rousseeuw et Leroy [19], consiste à minimiser la médiane des résidus au carré :

$$\min_{\mathbf{p}} \text{med}_i r_i^2. \tag{3}$$

En minimisant la médiane, on ignore la deuxième moitié des résidus triés par ordre croissant. Cette méthode est donc capable de prendre en compte des données comprenant jusqu'à 50% d'erreurs grossières. Cependant, comme seule une partie des données est utilisée, LMS n'est pas très précis. C'est pourquoi on l'affine souvent par une estimation aux *moindres carrés réduit* (RLS), c'est à dire une estimation aux moindres carrés incluant toutes les données dont le résidu est inférieur à un certain seuil :

$$\min_{\mathbf{p}} \sum_{i=1}^{n} w_i r_i^2, \text{ où } w_i = \begin{cases} 1 \text{ si } |r_i| \leq 2.5 \hat{\sigma}, \\ 0 \text{ si } |r_i| > 2.5 \hat{\sigma}. \end{cases} \tag{4}$$

$\hat{\sigma}$ est une approximation de l'écart type des erreurs résiduelles, et doit être lui-même estimé de façon robuste : on prend

$$\hat{\sigma} = 2.6477 \sqrt{\frac{1}{h} \sum_{i=1}^{h} r_{o(i)}^2}, \tag{5}$$

où $r_{o(1)}^2 \leq \ldots \leq r_{o(n)}^2$ sont les résidus au carré ordonnés, et h est égal à $n - [n/2]$. Tout comme LMS, le second facteur de l'équation 5 ne considère que la première moitié des résidus triés. Le facteur 2.6477 est introduit car $\frac{1}{h} \sum_{i=1}^{h} r_{o(i)}^2$ vaut approximativement $1/2.6477^2$ lorsque les résidus sont des variables aléatoires distribuées selon la loi de distribution normale $N(0,1)$ (voir [19] pour plus de détails). Malheureusement, malgré l'utilisation de RLS, cette méthode conduit fréquemment à un minimum local. En effet, comme LMS ne minimise que les résidus de la moitié des primitives, ceux-ci deviennent souvent bien plus petits que les résidus des autres primitives, qui ne sont donc pas prises en compte par RLS (voir figure 2 : les lignes pleines correspondent aux primitives 2D suivies, les lignes pointillées aux projections des correspondants 3D; les primitives 2D qui ne sont pas prises en compte dans l'optimisation - 5, 7 et 8 - apparaissent en noir : ici l'information de profondeur n'est pas représentée, ce qui implique un résultat très mauvais dans cette direction).
C'est pourquoi nous préférons utiliser la technique de M-estimation, développée par Huber [11], qui consiste à minimiser la somme d'une fonction des résidus :

$$\min_{\mathbf{p}} \sum_{i=1}^{n} \rho(r_i), \tag{6}$$

où ρ est une fonction continue et symétrique, ayant un minimum en zéro. Sa dérivée $\psi(x)$ est appelée *fonction d'influence*, car elle se comporte comme une fonction pondérante dans l'optimisation (6). Ces fonctions sont très efficaces, mais non adaptées à des situations comprenant plus de 20% (environ) de primitives aberrantes. La table 1 présente quelques fonctions d'influence couramment utilisées, avec leur représentation graphique. Parmi ces estimateurs, certains sont plus restrictifs que d'autres : alors que l'influence de la fonction de Tukey est nulle pour les résidus supérieurs à c, celle de Cauchy reste supérieure à zéro tout en décroissant, et celle de Huber reste constante égale à c. Pour cette raison, nous préférons utiliser Tukey, qui est suffisamment restrictif pour supprimer l'influence des primitives aberrantes, mais qui prend toutes les données en considération (par opposition à LMS). Le seuil c est pris égal à kS, où k est une constante (on prend $k = 4$ pour Tukey), et S un facteur d'échelle égal à $\hat{\sigma}$.

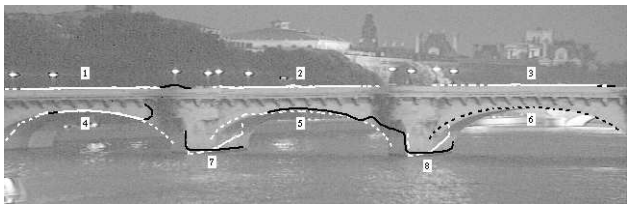
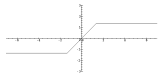
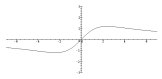
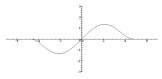


FIG. 2: *Utilisation de LMS au niveau global : ici l'information de profondeur (primitives 7 et 8) n'est pas prise en compte.*

La minimisation (6) peut être effectuée au moyen de techniques standards utilisant une estimation initiale de $\mathbf{p}$: une approche très simple comme la méthode

Nom	$\rho(x)$	$\psi(x)$	Graph. de $\psi(x)$
Huber $\begin{cases} \text{si } \lvert x\rvert \le c \\ \text{si } \lvert x\rvert > c \end{cases}$	$\begin{cases} x^2/2 \\ c(\lvert x\rvert - c/2) \end{cases}$	$\begin{cases} x \\ c * \mathrm{sgn}(x) \end{cases}$	
Cauchy	$\frac{c^2}{2}\log\left(1+\left(\frac{x}{c}\right)^2\right)$	$\frac{x}{1+\left(\frac{x}{c}\right)^2}$	
Tukey $\begin{cases} \text{si } \lvert x\rvert \le c \\ \text{si } \lvert x\rvert > c \end{cases}$	$\begin{cases} \frac{c^2}{6}\left[1-\left(1-\left(\frac{x}{c}\right)^2\right)^3\right] \\ c^2/6 \end{cases}$	$\begin{cases} x\left(1-\left(\frac{x}{c}\right)^2\right)^2 \\ 0 \end{cases}$	

TAB. 1: *Quelques M-estimateurs couramment utilisés.*

de Powell [17] s'est avérée suffisante dans notre cas et relativement rapide à l'exécution (dans le cadre du recalage temporel, l'estimée initiale de $\mathbf{p}$ est le point de vue calculé dans l'image précédente).

4.3 Le niveau local

Le but de ce niveau est de réduire l'influence des sections erronées des primitives. Pour cela, l'erreur résiduelle de la courbe C_i est calculée par une fonction robuste des distances $\{d_{i,j}\}_{1\le j\le l'_i}$. Nous pourrions prendre r_i égal à la médiane des $\{d_{i,j}\}$, mais là encore, cette méthode peut conduire à un minimum local où seule une partie de la primitive projetée est superposée avec la primitive 2D correspondante (voir figure 3). Nous préférons donc utiliser un M-estimateur, en prenant

$$r_i^2 = \frac{1}{l'_i}\sum_{j=1}^{l'_i}\rho(d_{i,j}). \qquad (7)$$

Cette optimisation ne doit pas être trop restrictive, pour la raison que nous venons d'évoquer : nous avons donc choisi d'utiliser Huber pour le niveau local (avec $k = 2$), ce qui s'est révélé être un bon choix dans notre expérimentation.

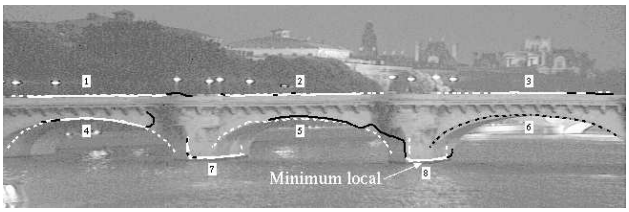


FIG. 3: *Utilisation de LMS au niveau local : on obtient un minimum local pour la primitive 8.*

4.4 Elimination des primitives aberrantes

Lorsqu'une première estimation du point de vue est obtenue par la méthode précédente, la détection des primitives aberrantes se fait de façon très simple : comme ces primitives ne devraient pas avoir influencé l'estimation finale, leurs résidus doivent être considérablement plus grands que les résidus des primitives correctes. Il suffit donc de comparer les résidus à leur écart-type : si $r_i > 2.5\,\hat{\sigma}$ (où $\hat{\sigma}$ est donné par l'équation 5), alors la primitive est sortie de l'ensemble $\mathcal{S}$ des primitives suivies, utilisées pour le calcul du point de vue. On peut ensuite affiner le point de vue en effectuant une estimation aux moindres carrés sur les primitives conservées (c'est-à-dire un LS au niveau global, r_i étant toujours donné par l'équation(7)).

5 Mise à jour des primitives

Au fur et à mesure que la caméra se déplace, de nouvelles primitives peuvent apparaître, tandis que d'autres peuvent disparaître. Il est donc essentiel de pouvoir mettre à jour dynamiquement l'ensemble $\mathcal{S}$ des primitives suivies dans la séquence.
Etant donné un point de vue calculé de façon robuste, nous sommes capables de déterminer les primitives 3D qui se projettent dans l'image. Si une nouvelle primitive apparaît, il faut alors déterminer la primitive 2D correspondante, qui sera suivie dans les images suivantes (nous nommerons cette étape *mise à jour d'une primitive*). La méthode utilisée pour mettre à jour une primitive est la suivante :

Etape 1 : on effectue une détection de contours classique (Canny [3]).

Etape 2 : on considère tous les contours qui sont suffisamment proches de la projection de la primitive 3D (par exemple les 10 premiers contours triés selon la taille croissante du résidu robuste obtenu par l'équation (7)).

Etape 3 : pour chacun de ces contours c, on calcule le point de vue robuste correspondant à $\mathcal{S} \cup c$, et on regarde si c est éliminé ou non. Si c est conservé, on

élimine les sections erronées de ce contour (en comparant les distances $d_{i,j}$ à leur écart type), et on stocke les sections conservées.

Etape 4 : si aucun contour n'est retenu, la mise à jour échoue, sinon on essaye de concaténer toutes les sections conservées et on retourne le plus long contour concaténé (la primitive image peut ainsi correspondre à l'union de plusieurs chaînes ou morceaux de chaînes).

Cette méthode est aussi utilisée pour mettre à jour les primitives éliminées durant la phase de calcul du point de vue, et pour déterminer dans la première image l'ensemble des primitives 2D à suivre (l'ensemble $\mathcal{S}$ contient alors initialement les quatre points utilisés pour calculer le point de vue initial - voir figure 5).

6 Expérimentation

Nous présentons dans cette section une application de notre méthode à un projet de réalité augmentée : l'illumination des ponts de Paris. L'objectif est de tester par synthèse plusieurs projets d'illumination d'un certain nombre de ponts situés autour de l'"Ile de la Cité". Il s'agit donc d'incruster l'image de synthèse du pont illuminé dans une séquence d'images réelles comprenant le pont non illuminé (et de voir l'effet de l'illumination sur les éléments environnants).
Une séquence panoramique de 300 images du Pont-Neuf a été prise à la tombée de la nuit à partir d'un autre pont. Les conditions d'expérimentation sont assez défavorables :

- le fait que les images soient sombres et bruitées a une forte influence sur la qualité de la segmentation (figure 6.a);
- la modélisation du pont a été effectuée à la main à partir de plans architecturaux. Le résultat, qui est très imprécis à certains endroits (voir notamment les arches sur la figure 4), pourrait être considérablement amélioré en utilisant une modélisation au laser par exemple;
- nous ne connaissons pas l'élévation de la surface du fleuve. Pour la déterminer, nous devons procéder de la façon suivante : on détermine dans la première image un ensemble de correspondances de points incluant des points à la surface du fleuve. Une première approximation h_0 de l'élévation peut être calculée car nous connaissons les dimensions du pont. On calcule ensuite le point de vue (par Dementhon) pour chaque élévation comprise dans l'intervalle $[h_0 - 50\text{cm}, h_0 + 50\text{cm}]$, et on garde l'élévation pour laquelle l'erreur de reprojection est minimale;
- les paramètres intrisèques de la caméra ont été obtenus par calibration sur un objet de référence proche de l'observateur (environ 3 mètres), alors que la scène considérée est beaucoup plus éloignée de la caméra (de l'ordre de 300 mètres), ce qui induit beaucoup de bruit sur ces paramètres.

Ces constatations devront être présentes à l'esprit lors de la visualisation des résultats.

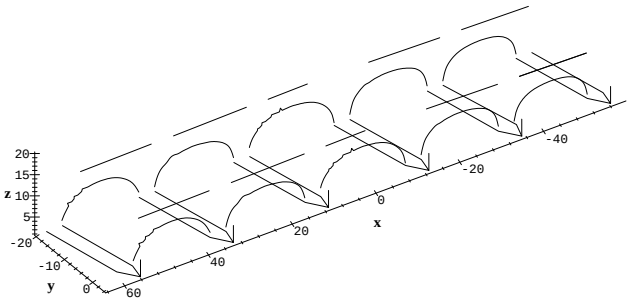


FIG. 4: *Le modèle en fil de fer du pont.*

L'étape d'initialisation est présentée en figure 5. Les conventions de dessin seront les même dans toute cette section : les lignes pleines correspondent aux primitives 2D suivies dans l'image, et les lignes pointillées aux projections des primitives 3D correspondantes (les projections des primitives qui ne sont pas - encore - mises à jour apparaissent en noir). La figure 5.a montre la projection des primitives 3D visibles dans l'image après calcul du point de vue par Dementhon sur les 4 points indiqués par des croix. La figure 5.b montre le résultat de la détection des primitives 2D correspondantes (les deux primitives de droite ne sont pas encore mises à jour car elles n'apparaissent pas entièrement dans l'image).

a.
b.

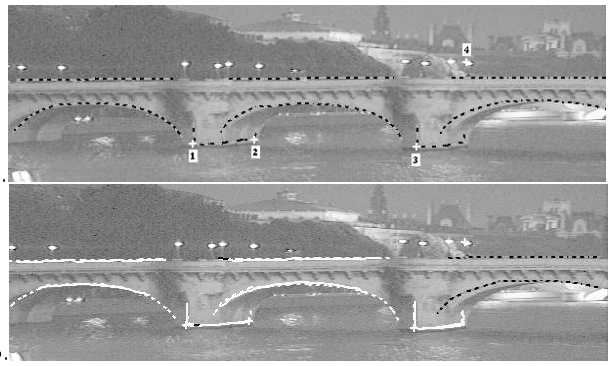


FIG. 5: *Etape d'initialisation. (a) Initialisation du point de vue par Dementhon à partir des 4 points désignés par des croix. (b) Détection automatique des primitives 2D à suivre.*

La figure 6.b montre les primitives 2D et la projection des primitives 3D obtenues dans la $12^{\text{ème}}$ image. On constate que la primitive 4 n'est pas entièrement correcte (ceci est principalement dû au fait que les *snakes* sont attirés par les gradients forts). Cependant, comme une partie de cette primitive correspond au modèle 3D, celle-ci est quand-même retenue. Le résultat du suivi dans la $13^{\text{ème}}$ image est présenté en figure 6.c. Exceptée la primitive 5, les autres primitives sont correctement suivies. L'erreur sur la primitive 5 est due à un échec de l'étape de prédiction en raison du bruit présent dans l'image.
La figure 6.d montre le résultat du calcul robuste du point de vue à l'issue du suivi (afin de mettre en évidence les parties des contours dont l'influence est réduite lors de l'estimation, nous avons tracé en noir les points pour lesquels le résidu est supérieur à c, c étant définie dans la table 1). Malgré les imprécisions évoquées ci-dessus, le résultat est visuellement convainquant, et la primitive 5, qui ne fausse pas le calcul du point de vue, est éliminée (les primitives aberrantes sont représentées en noir).
En figure 6.e, une primitive est mise à jour : il s'agit ici de la primitive 5 qui vient d'être éliminée. Pour finir, le point de vue est réestimé à partir du nouvel ensemble de primitives (la projection du modèle filaire ainsi obtenue apparaît en figure 6.f). Un résultat de composition finale est présenté en figure 7 (des précisions sur cette composition peuvent être trouvées dans [2]).

FIG. 7: *Composition finale pour l'image* 60.

Le lecteur intéressé est invité à visualiser trois séquences MPEG accessibles sur notre serveur web [1] : la première montre pour chaque itération les primitives 2D et la projection des primitives 3D obtenues à l'issue du calcul du point de vue (après le suivi et après la mise à jour des primitives), la deuxième la projection du modèle filaire à la fin de chaque itération, et la dernière un exemple de composition finale.
Quelques commentaires s'imposent ici : si l'on considère les conditions d'expérimentation évoquées plus haut, l'impression d'ensemble est assez convaincante. Nous montrons en figure 8 l'évolution du point de vue sur la séquence. Comme cette séquence a été filmée à partir d'une caméra posée sur un trépied, la translation devrait rester constante (figure 8.a,b,c - la translation est exprimée dans le repère du pont, voir la figure 4 pour les axes). On constate que la coordonnée en z est assez instable : ceci est principalement dû aux fortes imprécisions sur le modèle 3D qui affectent essentiellement la hauteur du pont, et à la détermination expérimentale de l'élévation de la surface du fleuve. Les angles d'euler pour la rotation sont représentés en figure 8.d,e,f. L'angle α, qui exprime la rotation autour de l'axe des z, évolue régulièrement alors que la caméra tourne. Les deux autres angles devraient être à peu près constants. Les écarts types des paramètres de la caméra devant être constants sur la séquence panoramique sont montrés en table 2.

Parametre	*Moyenne*	*Ecart type*
t_x (m)	-91.686634	1.678529
t_y (m)	325.359777	2.339981
t_z (m)	13.092605	2.547535
β (rad)	0.009910	0.005982
γ (rad)	-0.000236	0.005982

TAB. 2: *Moyenne et ecart type des paramètres de la caméra.*

La figure 9 montre l'évolution sur la séquence du nombre de primitives utilisées (pour les 128 premières images) : le premier niveau des barres (partie claire) indique le nombre de primitives utilisées pour le calcul du point de vue, et le second niveau le nombre de primitives visibles dans l'image (que l'on cherche à mettre à jour). La différence entre les deux niveaux (partie sombre) indique donc le nombre de primitives rejetées par le calcul du point de vue (baisse du niveau 1 avec niveau 2 constant) ou nouvellement présentes dans l'image, et dont la mise à jour a échoué. On voit que le nombre de primitives utilisées varie entre 3 et 7, tandis que le nombre de primitives visibles dans chaque image est compris entre 6 et 8 (5 primitives ont disparu et 6 autres sont apparues dans cet extrait de séquence). On observe deux longs tronçons (images 40 à 61 et 112 à 128) où seules 4 primitives sont utilisées alors que 7 sont visibles dans l'image : ceci est dû au fait que le recalage est très précis sur les 4 primitives considérées et rend donc difficile l'intégration d'une nouvelle primitive trop bruitée. On constate aussi qu'une primitive nouvelle est très rarement prise en compte immédiatement : ceci est dû à une légère distorsion de l'image dont l'effet est plus important sur les bords (zone où apparaîssent les primitives).

1. `http://www.loria.fr/~gsimon/sequence{1,2,3}.mpg`

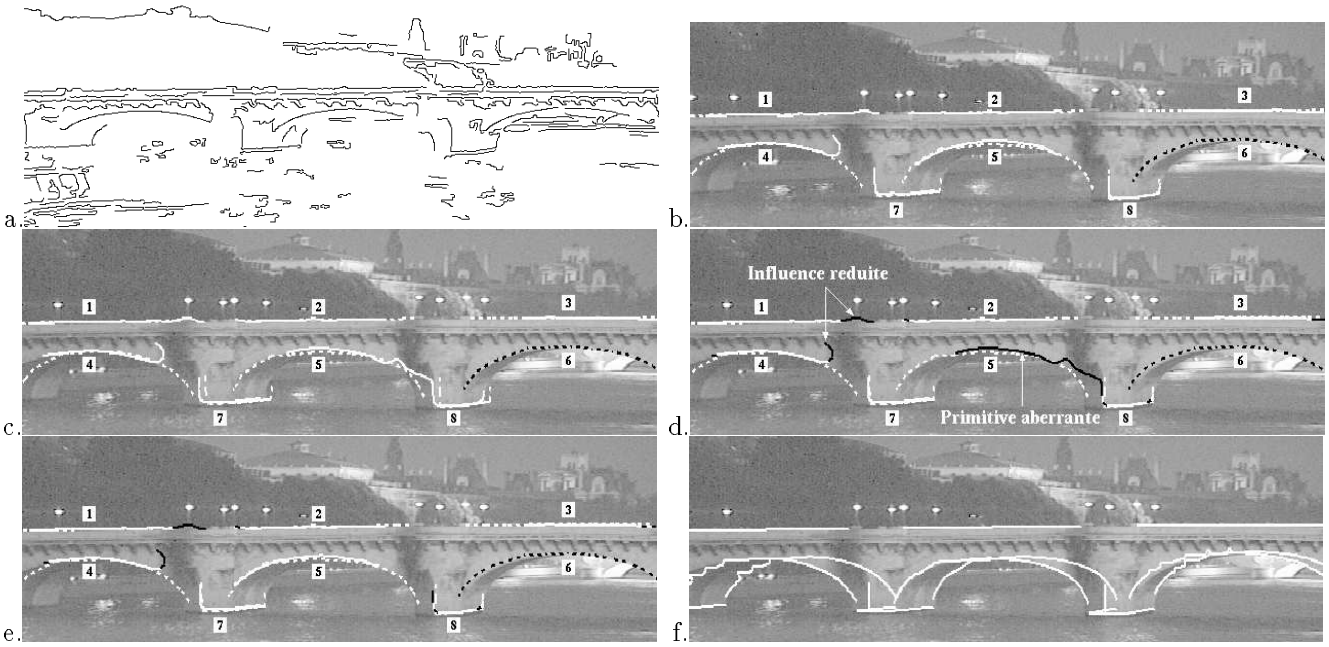


FIG. 6: *Recalage temporel pour la* $13^{ème}$ *image. (a) Chaînes de contours. (b) Primitives 2D avant le suivi (image 12). (c) Suivi dans l'image 13 (les projections des primitives 3D sont celles de l'image 12). (d) Calcul robuste du point de vue. (e) Mise à jour de l'ensemble des primitives utilisées. (f) Reprojection du modèle après la mise à jour.*

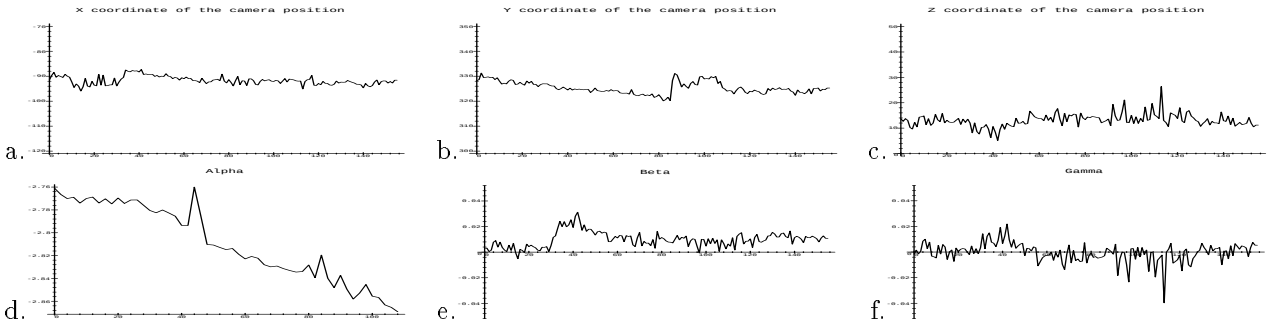


FIG. 8: *Evolution du point de vue (translation et angles d'euler) sur la séquence.*

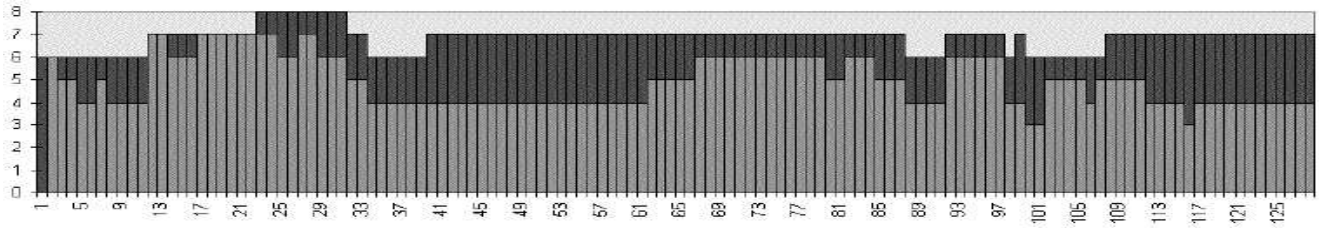


FIG. 9: *Evolution sur la séquence du nombre de primitives utilisées pour le calcul du point de vue (premier niveau) et du nombre de primitives visibles dans l'image (deuxième niveau).*

7 Conclusion

Nous avons présenté dans ce papier un système de recalage temporel capable de suivre dans une séquence d'images un objet 3D dont le modèle est connu. L'accent a été mis sur la méthode de calcul du point de vue, qui prend en compte des primitives quelconques de façon très robuste et est aussi utilisée pour mettre à jour l'ensemble des primitives suivies dans la séquence. Sa fiabilité a été montrée sur une séquence de 300 images d'un pont filmé à la tombée de la nuit, dans le cadre d'un projet de réalité augmentée.
Plusieurs directions peuvent être envisagées pour améliorer notre système : actuellement, nos algorithmes ne s'exécutent pas en temps réel. Ceci ne constituait pas une priorité jusqu'ici, puisque notre objectif était d'incruster des images de synthèse très réalistes dans une séquence pré-acquise (le processus de synthèse est plus coûteux que le recalage). Cependant, dans une optique temps réel, nous pourrions effectuer en parallèle sur chaque primitive aussi bien le suivi que le calcul des résidus. D'autre part, si des points ont été utilisés pour l'initialisation, seules des courbes ont été suivies dans la séquence. Puisque notre méthode est capable de les utiliser, nous pourrions envisager de suivre aussi des points (par un méthode de suivi de coins ou autre). Enfin, le modèle était ici structuré en un ensemble de primitives choisies par l'opérateur, mais ceci peut s'avérer contraignant lorsque le modèle est particulièrement complexe : nous travaillons actuellement sur une méthode d'extraction automatique des primitives 3D les plus pertinentes.

Références

[1] M.-O. Berger. How to Track Efficiently Piecewise Curved Contours with a View to Reconstructing 3D Objects. In *Proceedings of the 12th ICPR, Jerusalem (Israel)*, volume 1, pages 32–36, 1994.

[2] M.-O. Berger, C. Chevrier, and G. Simon. Compositing Computer and Video Image Sequences: Robust Algorithms for the Reconstruction of the Camera Parameters. In *Eurographics'96, Poitiers, France*, volume 15, pages 23–32, August 1996.

[3] J. Canny. A Computational Approach to Edge Detection. *IEEE Transactions on PAMI*, 8(6):679–698, 1986.

[4] D. Dementhon and L. Davis. Model Based Object Pose in 25 Lines of Code. *IJCV*, 15:123–141, 1995.

[5] M. Dhome, M. Richetin, J.T. Lapresté, and G. Rives. Determination of the Attitude of 3-D Objects from a Single Perspective View. *IEEE Transactions on PAMI*, 11(12):1265–1278, 1989.

[6] O. D. Faugeras and G. Toscani. The Calibration Problem for Stereo. In *Proceedings of IEEE Conference on CVPR, Miami, FL (USA)*, pages 15–20, 1986.

[7] J. Feldmar, N. Ayache, and F. Betting. 3D-2D projective registration of free form curves and surfaces. *Computer Vision and Image Understanding*, 65(3):403–424, 1997.

[8] M. Ferri, F. Mangili, and G. Viano. Projective Pose Estimation of Linear and Quadratic Primitives in Monocular Computer Vision. *CVGIP: Image Understanding*, 58(1):66–84, July 1993.

[9] D. Gennery. Visual Tracking of Known Three Dimensional objects. *IJCV*, 7(3):243–270, 1992.

[10] R. M. Haralick, H. Joo, C. N. Lee, X. Zhuang, V.G. Vaidya, and M. B. Kim. Pose estimation from corresponding point data. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(6), 1989.

[11] P. J. Huber. *Robust Statistics*. Wiley, New York, 1981.

[12] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active Contour Models. *IJCV*, 1:321–331, 1988.

[13] D. Koller, K. Daniilidis, and H. H. Nagel. Model-Based Object Tracking in Traffic Scenes. In *Proceedings of Second ECCV, Santa Margherita Ligure (Italy)*, volume 588 of *Lecture Notes in Computer Science*, pages 437–452, 1992.

[14] D. Kriegman and J. Ponce. On Recognizing and Positioning Curved 3D objects from Image Contours. *IEEE Transactions on PAMI*, 12(12):1127–1137, December 1990.

[15] R. Kumar and A. Hanson. Robust Methods for Estimating Pose and a Sensitivity Analysis. *CVGIP: Image Understanding*, 60(3):313–342, 1994.

[16] D. Lowe. Robust Model based Motion Tracking Trough the Integration of Search and Estimation. *IJCV*, 8(2):113–122, 1992.

[17] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C, The Art of Scientific Computing*. Cambridge University Press, 1988.

[18] S. Ravela, B. Draper, J. Lim, and R. Weiss. Tracking Object Motion Across Aspect Changes for Augmented Reality . In *ARPA Image Undertanding Worshop, Palm Spring (USA)*, August 1996.

[19] P. Rousseeuw and A. Leroy. *Robust Regression and Outlier Detection*. Wiley Series in Probability and Mathematical Statistics. Wiley, 1987.

[20] T. Shakunaga. Robust Line Based Pose Enumeration From a Single Image. In *Proceedings of 4th ICCV, Berlin (Germany)*, pages 545–550, 1993.

[21] M. Uenohara and T. Kanade. Vision based object registration for real time image overlay. *Journal of Computers in Biology and Medecine*, 1996.

[22] Z. Zhang, R. Deriche, O. Faugeras, and Q. Luong. A Robust Technique for Matching Two Uncalibrated Images Through the Recovery of the Unknown Epipolar Geometry. *Artificial Intelligence*, 78:87–119, October 1995.